(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 661 819 A2**

(12) EUROPEAN PATENT APPLICATION

(21) Application number: 94308662.9

(51) Int. Cl.[6]: H03M 7/30

(22) Date of filing: 23.11.94

(30) Priority: 29.12.93 US 174800

(43) Date of publication of application:
05.07.95 Bulletin 95/27

(84) Designated Contracting States:
DE FR GB

(71) Applicant: International Business Machines Corporation
Old Orchard Road
Armonk, N.Y. 10504 (US)

(72) Inventor: Jones, Ronald Henry
1130 Homewood Blvd.,
Bldg. F101
Delray Beach,
Florida 33445 (US)

(74) Representative: Lloyd, Richard Graham
IBM (UK) Ltd,
UK Intellectual Property Department,
Hursley Park
Winchester,
Hampshire SO21 2JN (GB)

(54) Data compression.

(57) Digitized audio information is compressed by retrieving a first audio sample from a digitized audio signal and storing it in its entire, uncompressed, form. The next audio sample is then retrieved and a signed difference between the first audio sample value and the next audio sample value is computed. If this difference value can be represented in fewer data segments than would be required to represent the next audio sample, the difference value is stored, rather than the value of the next sample, otherwise the next sample is stored. The invention proceeds in this fashion until the entire PCM audio signal is compressed and stored. The compressed data stream is decompressed to generate an output data stream by retrieving a stored sample and examining it to ascertain whether it is a difference value or a flag which indicates that the following data represents a coded amplitude value. If the sample is a difference value it is digitally added to the previous value (taking into account the sign) and the sum is inserted into the output data stream. The sum value is retained in order to be used to compute the next output value if the following sample is also a difference value. Alternatively, if the sample is a flag value it is discarded and the next sample is retrieved and inserted into the output data stream.

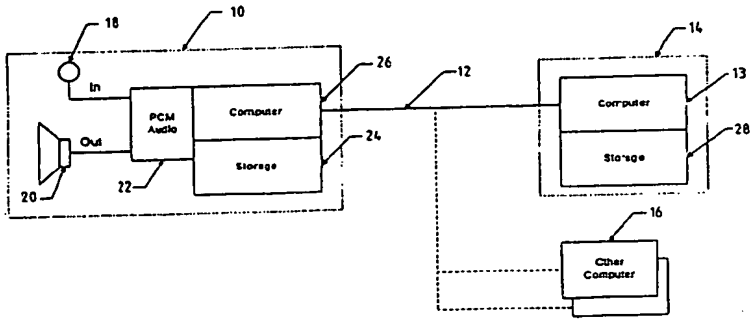

FIG. 1

EP 0 661 819 A2

This invention relates generally to the processing of digitized signals and more specifically to computer systems which compress and decompress coded digital data. such as pulse code modulated (PCM) audio signals.

The past decade has seen a revolution in computing. The advent and proliferation of personal computers has transformed the computing environment from one which was highly centralized and tightly controlled to one which is widely distributed with easy access. A significant expansion of computing applications is a concomitant result of this changing computer environment. In the past, computers provided, primarily, accounting, data reduction, and data-base management functions. They now, additionally, provide voice messaging, games, and multimedia applications, such as business presentations. While the older applications could be accommodated using only text-type data, the newer applications require graphical and audio data as well.

Graphical and, more to the point of this invention, audio signals require significant capacity for storage. For example, the word "hand" would require 4 bytes, 1 byte for each letter, for storage as text. On the other hand, the storage required for a digital audio version of "hand", assuming pulse code modulation (PCM) with 16 bits per sample with 20,000 samples per second, and assuming 1 second is required to utter "hand", is about 40,000 bytes. Although the cost-per-bit of computer storage has fallen dramatically, limited storage still imposes severe constraints on computer applications which use digitized audio. Consequently, it is highly desirable to compress digitized audio signals for use in multimedia computer environments.

Audio signals are commonly digitized using pulse code modulation (PCM) techniques. Pulse code modulation is applied by sampling an analog audio signal at a fixed rate, for example, 20 kHz, to produce a stream of pulse samples. The modulation technique then assigns a digital value to each sample which is representative of its amplitude.

Attempts have been made to compress PCM audio, but these attempts have generally required the addition of specialized compression/decompression equipment to existing computer equipment. The equipment typically receives PCM audio signals from an audio system, compresses the signals, and passes the compressed signals, as data, to a computer, which in turn stores the data. In order to regenerate the signals any system which retrieves the compressed data must also possess the specialized compression/decompression equipment.

The additional specialized compression/decompression equipment increases the cost and complexity of any system which employs it. Additionally, because the compression/decompression equipment generally compresses the PCM audio signal in a unique way, only other systems with compatible specialized equipment can utilize the compressed signals.

Some compression/decompression systems do not require specialized compression/decompression equipment. However, these systems do not provide real-time, lossless compression and decompression of high-quality PCM audio. They may require that PCM signals be stored and compressed off-line; a one-second audio signal would require more than one second to process. They may also provide lossy compression in order to obtain real-time operation. Lossy compression simply means that some of the signal's data is discarded in order to reduce the number of digits required to represent each sample.

This invention is directed to the provision of an improved compression/decompression scheme for digital data samples, particularly, though not exclusively, digital audio signals.

Accordingly, the invention provides apparatus for compressing a stream of coded digital data samples, the compressing apparatus comprising: comparison apparatus responsive to a first data sample value and a second consecutive data sample value in the stream for forming a difference value between the first data sample value and the second data sample value; and output apparatus for forming a compressed data stream, the output apparatus being responsive to the difference value and to the second data sample for inserting the difference value into the compressed data stream in place of the second data sample when the difference value occupies less storage space than the second data sample.

Also provided is apparatus for decompressing a stream of compressed coded digital data samples, the decompressing apparatus comprising: detection apparatus responsive to the compressed data stream for detecting a predetermined flag value; reconstruction apparatus cooperating with the detection apparatus for reconstructing an uncompressed data sample from compressed data samples when the flag value is not detected; and output apparatus for forming a decompressed data stream, the output apparatus being responsive to a detected flag value for inserting the next consecutive data sample into the decompressed data stream, the output apparatus inserting the reconstructed uncompressed data sample into the decompressed data stream when the flag value is not detected.

This enables digitized audio signals to be compressed for computer system storage in a way that eliminates the need for specialized compres-

BNSDOCID: <EP 0661819A2>

sion/decompression equipment, and with sufficient efficiency to permit real-time, lossless compression and decompression of high-quality digitized audio signals.

In accordance with one embodiment of the invention, digitized audio information is compressed by retrieving a first audio sample from a digitized audio signal and storing it in its entire, uncompressed, form. The next audio sample is then retrieved and a signed difference between the first audio sample value and the next audio sample value is computed. If this difference value can be represented in fewer data segments than would be required to represent the next audio sample, the difference value is stored, rather than the value of the next sample, otherwise the next sample is stored. The next audio sample is then retrieved, the signed difference value between this and the prior sample computed, and the difference value or the sample value is stored if it occupies less storage space, otherwise the sample value is stored along with a unique, flag, or key value which indicates that the sample is stored in an uncompressed form. The process proceeds in this fashion until the entire PCM audio signal is compressed and stored.

Because sequential audio samples are likely to be relatively similar in amplitude, the difference in amplitude from one sample to another is likely to consume significantly less storage space than the digitally-coded amplitude of the subsequent sample. Therefore, a high proportion of the compressed data will be stored as difference values. However, there will generally always be some samples stored as coded amplitudes. This occurs because digital data is stored and retrieved in fixed-size units and thus the coded amplitude of a sample may be stored even if the amplitude difference between two sequential samples is less than the coded amplitude. For example, audio signals are commonly digitized in 16-bit samples, but digital values are commonly stored in eight-bit bytes. Consequently, if the difference between two sample amplitudes is greater than can be represented in one byte, the amplitude value of the subsequent sample will be stored because the difference value will not occupy less storage space. A flag will also be stored to indicate that the value stored is anuncompressed sample. Alternatively, if the amplitude difference can be represented in one byte, then it will occupy less storage space than the coded amplitude value and the value stored for the sample will occupy one byte.

Consequently, when a sample amplitude is stored as a coded amplitude rather than a difference value, in a preferred embodiment. a "flag" is inserted into the compressed data stream which specifies that the following value is an amplitude value rather than a difference value. For example, in the case where two bytes are used to code an amplitude value, the flag specifies that the following two bytes represent a coded amplitude value (a flag is not utilized for the first sample however).

The compressed data stream is decompressed to generate an output data stream by retrieving the first stored sample from, for example, a storage device. The first sample is always a coded amplitude sample and is directly inserted into the output data stream. The next sample is retrieved and examined to ascertain whether it is a difference value or a flag which indicates that the following data represents a coded amplitude value. If the sample is a difference value it is digitally added to the previous value (taking into account the sign) and the sum is inserted into the output data stream. The sum value is retained in order to be used to compute the next output value if the following sample is also a difference value. Alternatively, if the sample is a flag value it is discarded and the next sample is retrieved and inserted into the output data stream. Operation continues in this way until all of the stored compressed data has been decompressed.

Viewed from another aspect, the invention provides a method for compressing a stream of coded digital data samples, the method comprising the steps of:

A. forming a difference value between the first data sample value and the second data sample value;

B. comparing the difference value to the second data value; and

C. forming a compressed data stream by inserting the difference value into thecompressed data stream in place of the second data sample when the difference value occupies less storage space than the second data sample, and

a method for decompressing a stream of compressed coded digital data samples, the method comprising the steps of:

A. monitoring the compressed data stream for a predetermined flag value;

B. reconstructing an uncompressed data sample from compressed data samples when the flag value is not detected; and

C. forming a decompressed data stream by inserting the next consecutive data sample into the decompressed data stream when the flag value is detected and by inserting the reconstructed uncompressed data sample into the decompressed data stream when the flag value is not detected.

The above and further advantages of the invention may be better understood by referring to the following description of the embodiment thereof in conjunction with the accompanying drawings, in which:

FIG. 1 shows an illustrative networked multimedia computer system;

FIGs. 2A-2C, when placed together, are a flow diagram which illustrates a digitized audio compression method.

FIGs. 3A-3C, when placed together, are a flow diagram which illustrates a method of decompressing audio data that has been compressed using the method illustrated in Figs. 2A-2C.

Referring to FIG. 1, a multimedia computer system 10 communicates over a network 12 with a computer system 14 and additional other computers 16. The precise characteristics of the network 12 are not important for the implementation of the present invention. For example, the network 12 may be implemented by any one of a variety of physical media, may exhibit any of a variety of topologies such as star or ring topologies and may be a local area network or wide area network. Computer systems 10, 14 and 16 transfer information between themselves through the network 12 in the form of digital data which may represent textual information, graphical information, or digitized audio information.

The illustrative multimedia computer system 10 includes a conventional audio input device 18 (such as a microphone), a conventional audio output device 20 (such as a speaker), and known PCM audio coding equipment 22. Equipment 22 may be a stand-alone unit or a plug-in card with direct communication to an internal bus of computer 26. A storage device 24 is also associated with computer system 10 and may be an integral part of computer 26 or it may be a stand-alone storage system.

Similarly, the storage device 28 may be an integral part of the computer 13 or a stand-alone device. Although not illustrated, computer system 14 may also have PCM audio coding equipment and audio input and output devices such as devices 18, 20 and 22 illustrated in conjunction with computer system 10. Alternatively, the computer system 14 may act as a server, for example, to store and retrieve data on the storage device 28 which may have sufficient capacity to providestorage for a number of computers attached to the network 12.

In operation, the conventional PCM audio coding equipment 22 accepts an analog audio signal from the input device 18. The analog audio signal is sampled at a fixed rate, for example, 20 kHz, to produce a stream of samples with varying analog amplitudes. The amplitude of each pulse is then quantized and the quantized value is coded into a binary code. The resulting PCM binary data stream is sent through computer 26 to storage device 24. Alternatively, the computer 10 may send the digital data over the computer network 12 to computer 14 for remote storage in storage unit 28 or to other computers 16 for additional processing.

The data compression method may be performed by software running on computer 26 or may be performed by dedicated hardware located in computer system 26. In either case, the PCM digital signal received from the PCM audio equipment 22 is received by computer system 26 and compressed prior to storage in the storage device 24 or transmission over network 12. Conversely, the data decompression method operates in conjunction with the computer 26 to retrieve and decompress audio data stored in the storage device 24 or transmitted over network 12 from storage 28 to produce a PCM digital data stream. The computer 26 transfers this PCM digital data stream to the PCM audio equipment 22 which converts the digital stream into an analog audio signal which is then transmitted to output device 20. Note that, in contrast other PCM data reduction systems, PCM digital audio data is produced for presentation to the PCM audio equipment 22 or to the network 12 (of course, compressed samples may also be presented to the network). As a result, no specialized digital audio equipment is required for use with the multimedia computer audio system 10 or with the other computer systems 14 or 16.

FIGs. 2A-2C form a flowchart illustrating the operation of the PCM audio compression scheme. In this exemplary embodiment it is assumed that the digitized audio samples arestored as eight-bit bytes and that each sample is sixteen bits, or two bytes, wide. The operation begins at step 200 and proceeds to step 210 where an output byte pointer variable, pout, and an input byte pointer variable, pIn, are initialized. These pointers point to storage locations in a data storage apparatus which holds the digitized audio samples during processing or to a location in a stream buffer used to transport the data. The value of the first input sample is transferred to a holding variable (InSamp) which represents the "next" input sample.

Next, at step 212, the routine determines whether the next input sample (stored in InSamp) is the first input sample of the digitized audio signal. If the next sample is the first sample of the signal, the invention proceeds to step 214 where it copies the value held in the next sample variable in a storage location (SCur) reserved for the "current" sample value. The current sample value will be used, as hereinafter explained, to determine the difference value between the current sample and the next sample.

The method then proceeds to step 216 where the input pointer (pIn) is incremented to indicate the location of the next input sample. As indicated in step 216, the input pointer is incremented by two since each sample comprises two bytes. The routine proceeds to step 218 where the value of the

BNSDOCID: <EP 0661819A2 >

next sample variable is stored in a storage location indicated by the output pointer (pOut) which points to the output data stream. At step 220, the output pointer is incremented, again by two to indicate that two bytes have been transferred to the output data stream.

The routine then proceeds to step 222, where a determination is made whether there is another sample in the digitized audio signal. If there are no more samples, the routine proceeds to step 224 to finish.

Returning to step 222, if there is another digitized audio input sample to be processed, theroutine proceeds to step 226, where it retrieves the next input sample by transferring the contents of the storage location identified by the input pointer (pIn) into the next sample holding variable, InSamp. Next the routine proceeds to step 228 where the input pointer is incremented by two so that the input pointer points to the next sample.

The routine then proceeds back to step 212, where it determines whether the value stored in the next sample variable InSamp is the first input sample or not. If the next sample is not the first input sample, the routine proceeds to step 230 where the difference between the "next" sample value contained in variable InSamp and the value of the current sample which is stored in the variable SCur. The value of this difference is stored in a variable, SDiff. At this time, a character output buffer, chOut is cleared (set equal to zero).

The routine then proceeds to step 232 where the current sample value stored in variable, SCur, is updated by transferring the value stored in the next sample variable InSamp. At step 234, the inventive method determines whether the difference between the next sample value and the current sample value is negative. If the difference is negative (indicating that the amplitude decreased) the routine proceeds to step 236 where the high-order bit within the character output buffer, chOut, is set to one (by loading the character buffer with 10000000, hex 80) to indicate a negative difference.

If, on the other hand, in step 234, the computed difference, SDiff, is equal to or greater than zero, the routine proceeds from step 234 directly to step 238, skipping step 236. The high-order bit thus remains at zero (it was cleared to zero in step 230). In step 238, the routine stores the absolute value of the difference SDiff in the location SDiff (the sign of the difference is now indicated by the high-order bit of chOut).

At step 240, the routine determines if either of the conditions (a) the absolute value of the computed difference SDiff is greater thanthe maximum positive value that can be stored in a single signed byte (01111111, hex 7F) or (b) the absolute value of the difference SDiff is equal to a reserved flag value (01111111, hex 7F with high-order byte of character buffer set; chOut = hex(80)) is true.

If neither condition is true, the difference SDiff between the next input sample and the current input sample can be stored in a single byte, and, in accordance with the principles of the present invention, the difference value is stored in the output data stream in place of the actual sample value. Accordingly, the routine proceeds to step 250 where the character buffer is loaded with the difference value by ORing the difference value SDiff with the contents of the character buffer (which may have the high-order bit set).

At step 252 the contents of the character buffer chOut are transferred to the output data stream location pointed to by output pointer pOut and, at step 254, the routine increments the output pointer pOut by one to indicate that a single byte has been transferred.

Returning to step 240, if the computed magnitude difference between the next and current data sample values was greater than the maximum value (hex 7F) or was equal to hex 7F with chOut equal to hex 80, the difference between the next sample and the current sample cannot be stored in a single byte and the actual value of the next sample will be stored in the output data stream.

Accordingly, the routine proceeds to step 242 where a flag value (hex FF) is written into the output data stream location indicated by the output pointer, pOut. The flag value, hex FF; indicates that the next two values in the data stream hold an uncompressed PCM audio sample value. At step 244, the routine increments the output pointer pOut by one to account for the flag value, then proceeds to step 246 where the two bytes of the next sample value within the variable InSamp is transferred to the output datastream to a location pointed to by the output pointer, pOut. The routine then proceeds to step 248, where the output pointer pOut is incremented by two to indicate that two bytes have been transferred.

Next, the routine proceeds to step 222 where it determines whether there is another input sample to be processed. The routine then proceeds, as previously described, either to step 226 to process more samples or to finish at step 224.

FIGs. 3A-3C form a flowchart detailing the steps in an illustrative decompression routine. More particularly, the routine begins the decompression process at step 300 then proceeds to step 310 where it initializes the output pointer, pOut, the input pointer, pIn and transfers the first sample of the compressed data to the next sample variable, InSamp. At step 312, a check is made to determine whether the next sample is the first sample of the compressed data. If it is the first sample, the rou-

tine proceeds to step 314 where it copies the value contained within the variable InSamp to the current sample variable SCur. At step 315, the input pointer, pIn, is incremented by two to account for the two bytes of the first sample

At step 316, the routine transfers the value contained within the location "InSamp" into the output stream location pointed to by the output pointer, pOut. At step 318, the output pointer, pOut, is incremented by two to indicate that two bytes have been output. Then, at step 320, the routine determines whether there are more input samples. If there are no more input samples, the routine proceeds to finish in step 322.

If there are more input samples, the process proceeds to step 324 where the next compressed sample byte stored in the location indicated by the input pointer, pIn, is read into an input buffer "chKey" so that the byte can be examined to determine if it is a flag value. At step 326, the input pointer, pIn, is incremented by one to account for the byte just read.

At step 312, the routine, upon determining that this is not the first sample, proceeds to step 328, where it clears the output character buffer "chOut" and proceeds to step 330. At step 330, the routine determines whether the contents of the input buffer "chKey" are equal to the flag value FF (hex). If the contents of chKey equal FF (hex), the next two bytes in the input data stream comprise an uncompressed PCM sample. Consequently, at step 332, the next input sample value from the input stream pointed to by the pointer, pIn, is transferred in the next sample variable. InSamp and the input pointer, pIn is incremented, at step 334, by two to account for the sample.

Returning to step 330, if the value within the input buffer, chKey, is not a flag value, then it is a difference value and the routine proceeds to step 336. In step 336 the high-order bit of the difference value is checked by ANDing contents of the input buffer, chKey and 10000000 (80 hex) and determining whether the resultant is equal to zero. As previously described, the high-order bit determines whether the compressed sample should be added to or subtracted from the previous sample value.

If the resultant of the AND operation is non-zero, indicating that the high-order bit is set and the difference is negative and should be subtracted from the current sample value in SCur, the routine proceeds to step 338 where the high-order bit within "chKey" is cleared (set to zero) by ANDing the contents of chKey and 01111111 (7F, hex). At step 340, the difference value in chKey is subtracted from the current input value in SCur. The resultant is transferred to the next sample variable, InSamp.

If the resultant was equal to zero at step 336, the high-order bit was clear indicating that the difference value is positive and should be added to the current value. Accordingly, in step 342, the current sample value in SCur is added to the difference value in chKey and the sum transferred to the next sample variable InSamp.

After either of steps 340, 342 or 334, the illustrative routine proceeds to step 344 whereit copies the next sample value in variable InSamp to the location indicated by the output pointer pOut, thereby placing the decompressed sample in the output data stream. At step 346 the output pointer, pOut, is incremented by two bytes and at step 348, the value within the variable InSamp is copied into the current sample variable SCur in preparation for processing the next sample.

Following step 348, the routine proceeds to step 320 where, as previously described, a determination is made whether there are more samples. Based on that decision, the additional samples are processed or the routine finishes if all samples have been processed.

As is clear from the foregoing description, although fewer digits are required to represent the signal in compressed form, the compression and decompression processes store and retrieve all the information contained within the original digitized signal. No information is discarded; this is a lossless compression/decompression process.

The foregoing description has set out a specific embodiment of this invention. It will be apparent, however, that variations and modifications may be made to the invention, with the attainment of some or all of its advantages. Therefore, it is the object of the appended claims to cover all such variations and modifications as come within the scope of the invention.

There has been described a data compression system for use with a stream of coded digital data samples each having a first predetermined number of data bytes, the compression system comprising: a computing device responsive to a first data sample value and to a consecutive second data sample value for computing a signed difference value between the first sample value and a second sample value, the difference value having a second number of bytes; and an output device for generating a compressed data byte stream, the output device being responsive to the difference value for inserting the difference value into the compressed data byte stream when the second number of bytes is less than the first number of bytes and for inserting the second data sample value into the compressed data stream when the second number of bytes is equal to, and greater than, the first number of bytes.

BNSDOCID: <EP 0661819A2>

As described, the system can comprise means for inserting a predetermined flag value into the compressed data stream prior to inserting the second data sample value into the compressed data stream when the second number or bytes is equal to, and greater than, the first number of bytes, and a storage mechanism for temporarily storing each data sample in the digital data stream. The computing device can be responsive to a value of a data sample stored in the storage mechanism and to a consecutive second data sample value for computing a signed difference value between the stored sample value and a second sample value.

Also, there has been described a data decompression system for decompressing a stream of compressed coded digital data samples, the decompressing system comprising: storage means for storing each compressed data sample; detection apparatus responsive to the stored data sample for detecting a predetermined flag value; reconstruction apparatus cooperating withthe detection apparatus for reconstructing an uncompressed data sample from the stored data sample and a subsequent consecutive compressed data sample when the flag value is not detected; and output apparatus for forming a decompressed data stream, the output apparatus being responsive to a detected flag value for inserting the next consecutive data sample into the decompressed data stream, the output apparatus inserting the reconstructed uncompressed data sample into the decompressed data stream when the flag value is not detected.

The reconstruction apparatus can comprise a mechanism for digitally adding the stored data sample to a subsequent compressed data sample to reconstruct the uncompressed data sample. Each compressed data sample can be comprised of a first number of data bytes, the difference value comprises a second number of bytes and wherein the storage mechanism temporarily stores the first number of bytes. The output apparatus can be responsive to a detected flag value for inserting the first number of consecutive data bytes into the decompressed data stream, the output apparatus inserting a signed addition of the second number of data bytes and the stored data bytes into the decompressed data stream when the flag value is not detected.

Claims

1. Apparatus for compressing a stream of coded digital data samples, the compressing apparatus comprising:

   comparison apparatus responsive to a first data sample value and a second consecutive data sample value in the stream for forming a difference value between the first data sample value and the second data sample value; and

   output apparatus for forming a compressed data stream, the output apparatus being responsive to the difference value and to the second data sample for inserting the difference value into the compressed data stream in place of the second data sample when the difference value occupies less storage space than the second data sample.

2. Apparatus as claimed in Claim 1 further comprising flag apparatus responsive to the difference value and to the second data sample for inserting a predetermined flag value into the compressed data stream when the difference value employs the same number of digits to represent the second data sample as the second data sample occupies.

3. Apparatus as claimed in Claim 1 or Claim 2 wherein each data sample is comprised of a first number of data bytes and wherein the comparison apparatus comprises apparatus for forming a digital difference value between the first data sample value and the second data sample value, the digital difference value comprising a second number of data bytes.

4. Apparatus as claimed in Claim 3 wherein the output apparatus comprises apparatus which inserts the difference value into the compressed data stream in place of the second data sample when the second number of bytes is less than the first number of bytes.

5. Apparatus as claimed in any preceding claim wherein the difference value includes a sign value which indicates whether the difference value ispositive or negative.

6. Apparatus for decompressing a stream of compressed coded digital data samples, the decompressing apparatus comprising:

   detection apparatus responsive to the compressed data stream for detecting a predetermined flag value;

   reconstruction apparatus cooperating with the detection apparatus for reconstructing an uncompressed data sample from compressed data samples when the flag value is not detected; and

   output apparatus for forming a decompressed data stream, the output apparatus being responsive to a detected flag value for inserting the next consecutive data sample into the decompressed data stream, the output apparatus inserting the reconstructed uncompres-

sed data sample into the decompressed data stream when the flag value is not detected.

7. Apparatus as claimed in Claim 6 further comprising a storage mechanism for storing a compressed data sample and wherein the output apparatus comprises a mechanism for inserting a signed addition of the next consecutive data sample and the stored data sample into the decompressed data stream when the flag value is not detected.

8. Apparatus as claimed in Claim 6 or Claim 7 wherein each compressed data sample is comprised of a first number of data bytes, the difference value comprises a second number of bytes and wherein the storage mechanism temporarily stores the first number of bytes.

9. Apparatus as claimed in Claim 8 wherein the output apparatus is responsive to a detected flag value for inserting the first number of consecutive data bytes into the decompressed data stream, the output apparatus inserting a combination of the second number of data bytes and the stored data bytes into the decompressed data stream when the flag value is not detected.

10. A method for compressing a stream of coded digital data samples, the method comprising the steps of:

A. forming a difference value between the first data sample value and the second data sample value;

B. comparing the difference value to the second data value; and

C. forming a compressed data stream by inserting the difference value into thecompressed data stream in place of the second data sample when the difference value occupies less storage space than the second data sample.

11. A method for decompressing a stream of compressed coded digital data samples, the method comprising the steps of:

A. monitoring the compressed data stream for a predetermined flag value;

B. reconstructing an uncompressed data sample from compressed data samples when the flag value is not detected; and

C. forming a decompressed data stream by inserting the next consecutive data sample into the decompressed data stream when the flag value is detected and by inserting the reconstructed uncompressed data sample into the decompressed data stream when the flag value is riot detected.

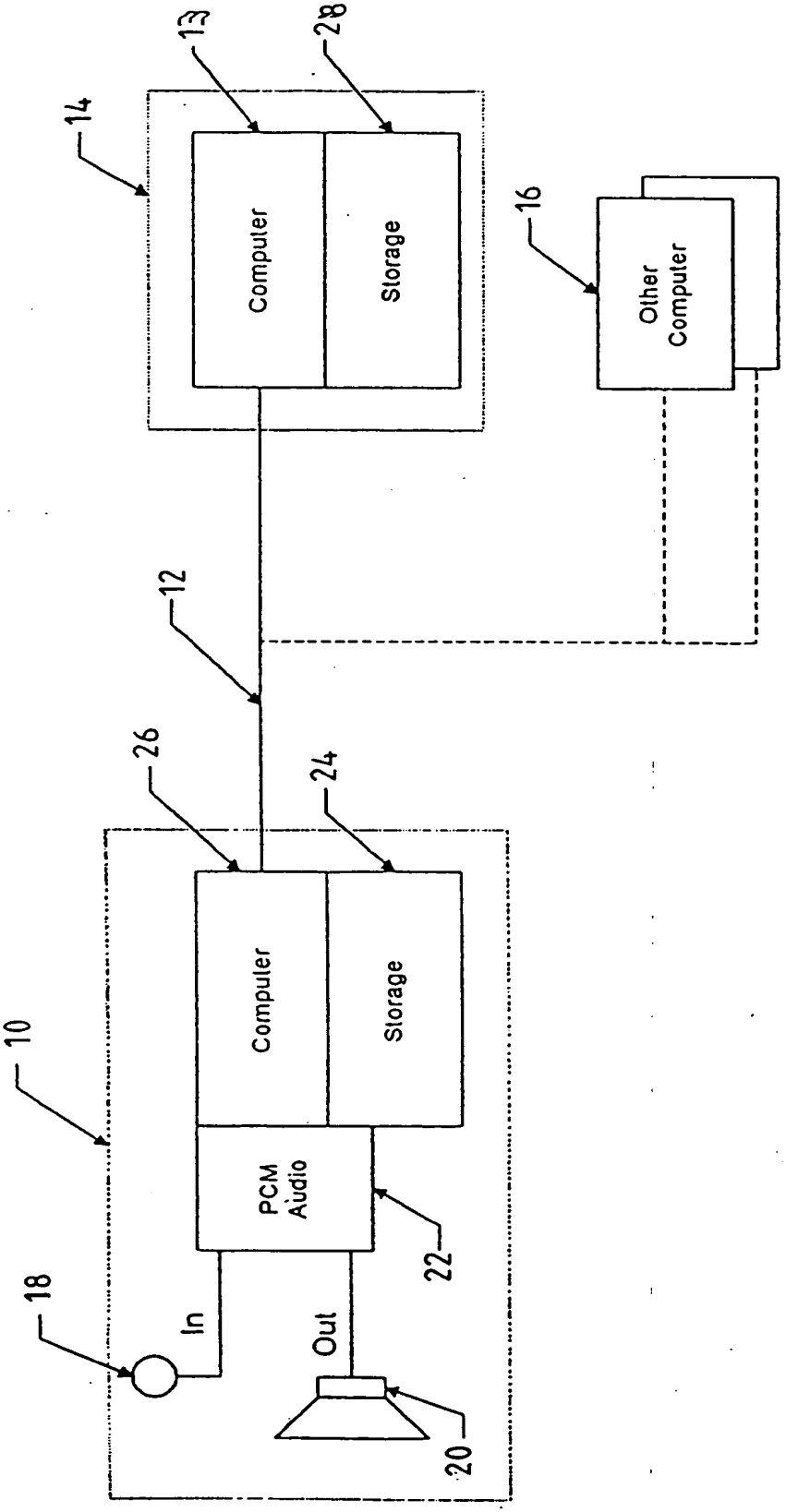
13
28
14
Computer
Storage
16
Other Computer
12
26
24
Computer
Storage
10
PCM Audio
22
18
In
Out
20

FIG. 1

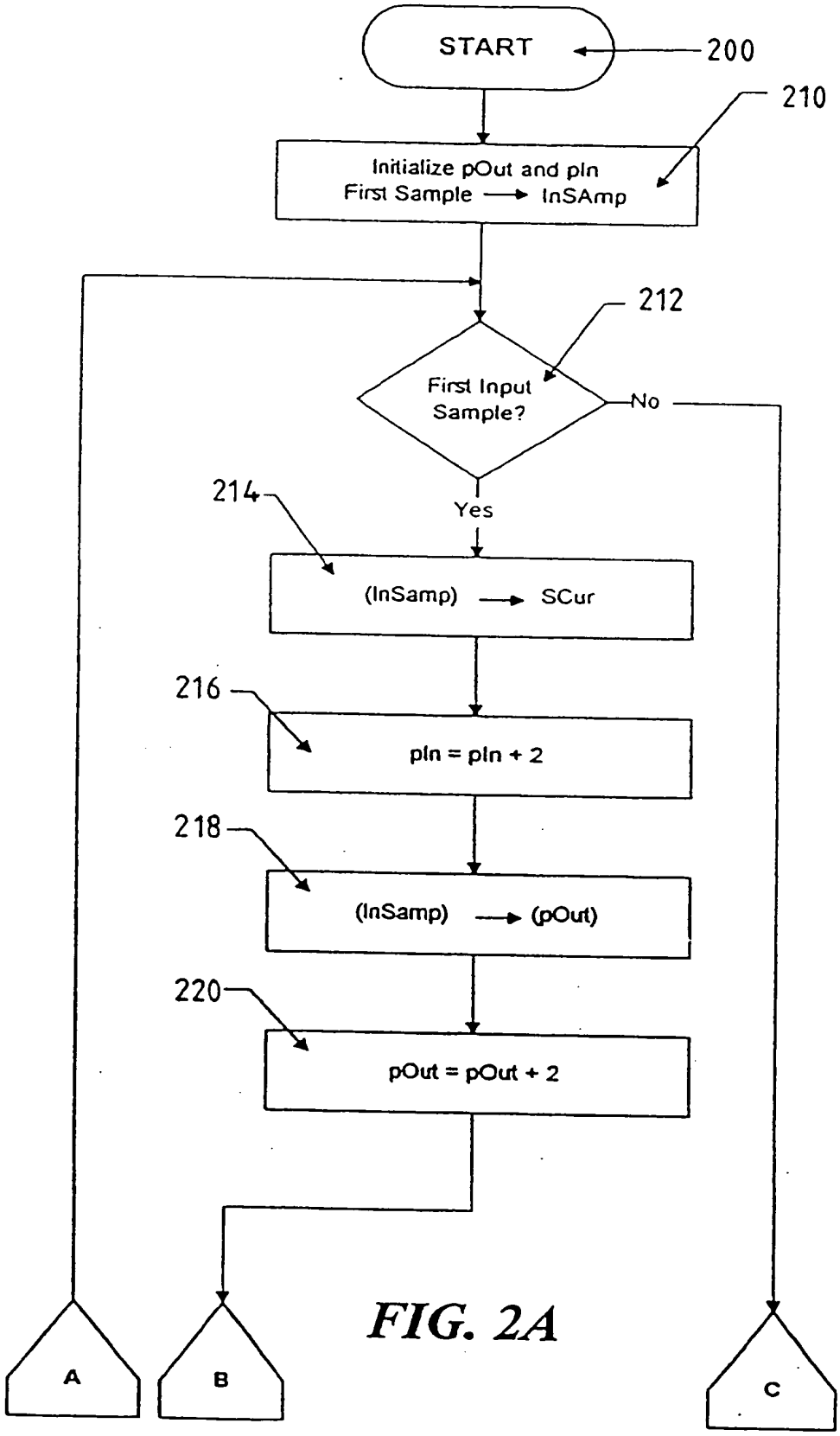
START
200
210
Initialize pOut and pIn
First Sample → InSAmp
212
First Input Sample?
No
Yes
214
(InSamp) → SCur
216
pIn = pIn + 2
218
(InSamp) → (pOut)
220
pOut = pOut + 2
A
B
C

FIG. 2A

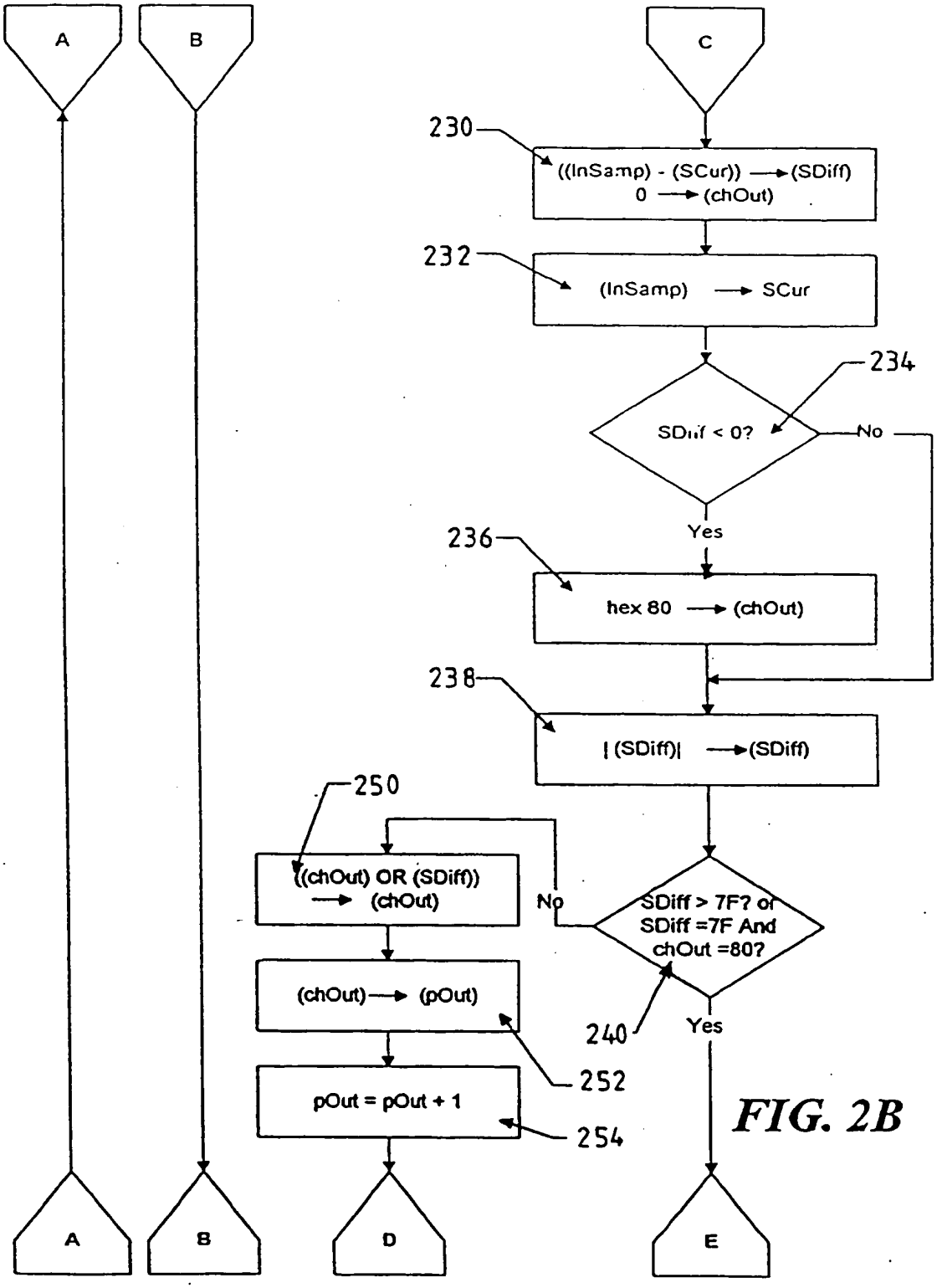
A
B
C
230
((InSamp) - (SCur)) → (SDiff)
0 → (chOut)
232
(InSamp) → SCur
234
SDiff < 0?
No
Yes
236
hex 80 → (chOut)
238
| (SDiff)| → (SDiff)
250
((chOut) OR (SDiff)) → (chOut)
No
SDiff > 7F? or SDiff =7F And chOut =80?
240
Yes
(chOut) → (pOut)
252
pOut = pOut + 1
254
A
B
D
E

FIG. 2B

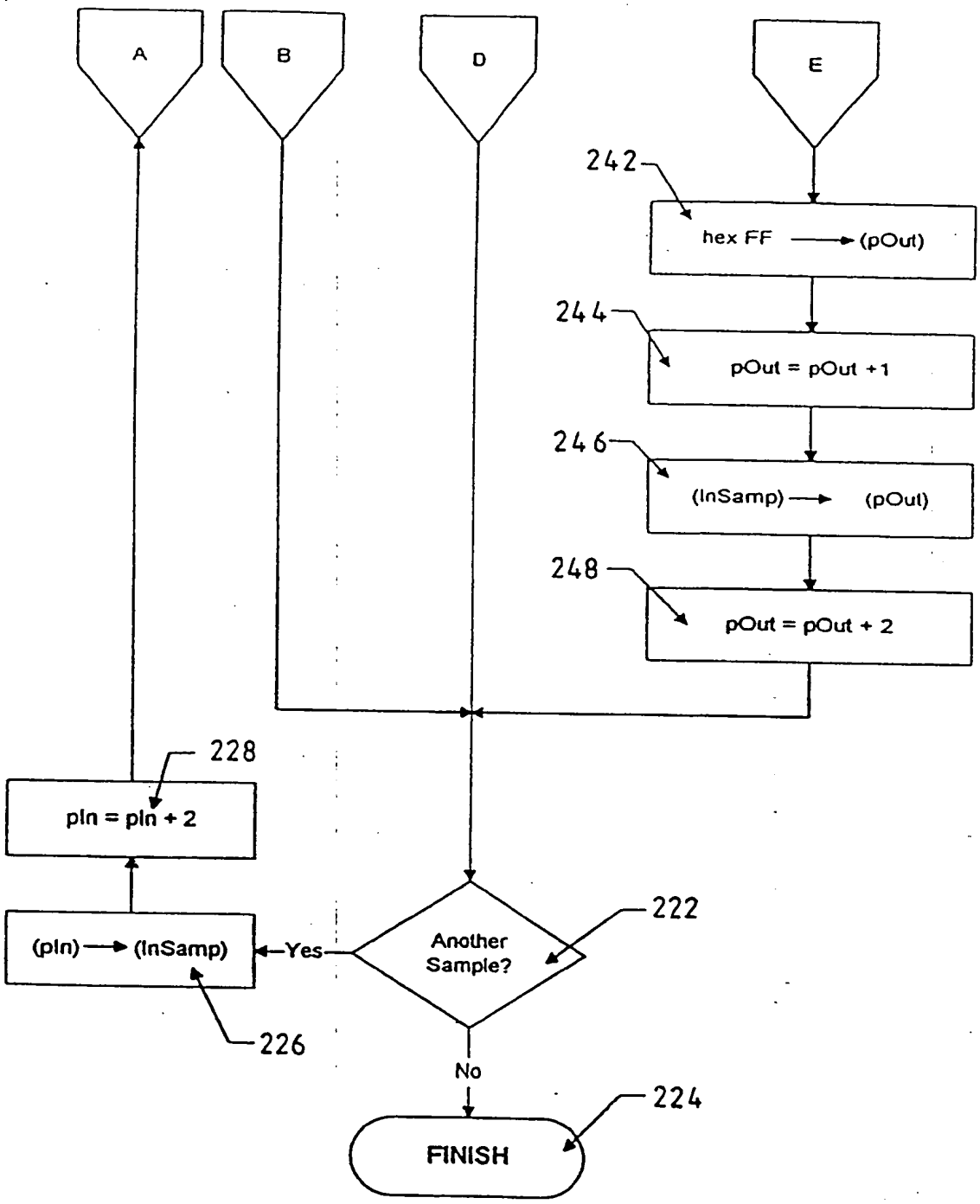
A
B
D
E
242
hex FF → (pOut)
244
pOut = pOut +1
246
(InSamp) → (pOut)
248
pOut = pOut + 2
228
pIn = pIn + 2
(pIn) → (InSamp)
Yes
Another Sample?
222
226
No
224
FINISH

FIG. 2C

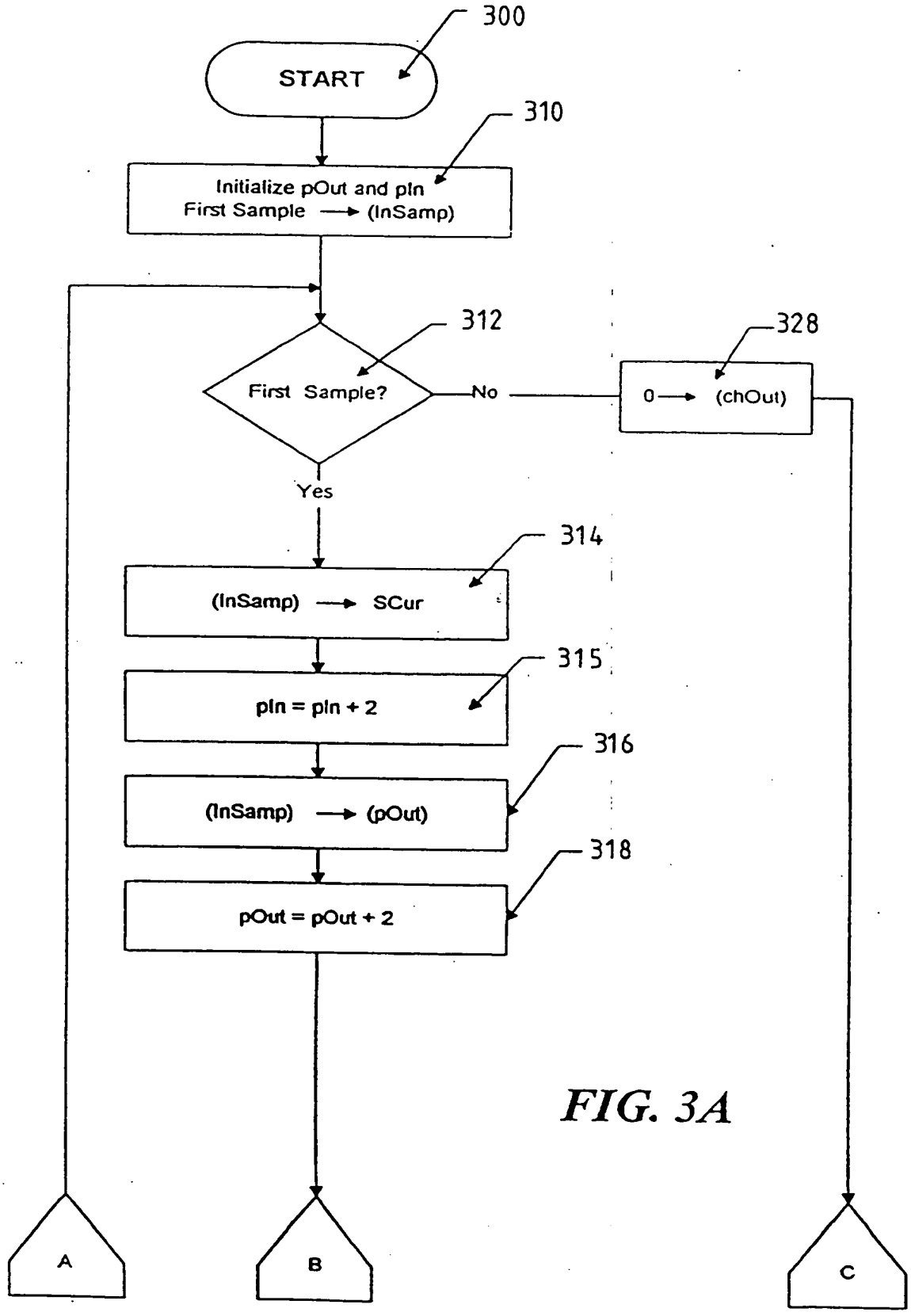
300
START
310
Initialize pOut and pIn
First Sample → (InSamp)
312
First Sample?
No
328
0 → (chOut)
Yes
314
(InSamp) → SCur
315
pIn = pIn + 2
316
(InSamp) → (pOut)
318
pOut = pOut + 2
A
B
C

FIG. 3A

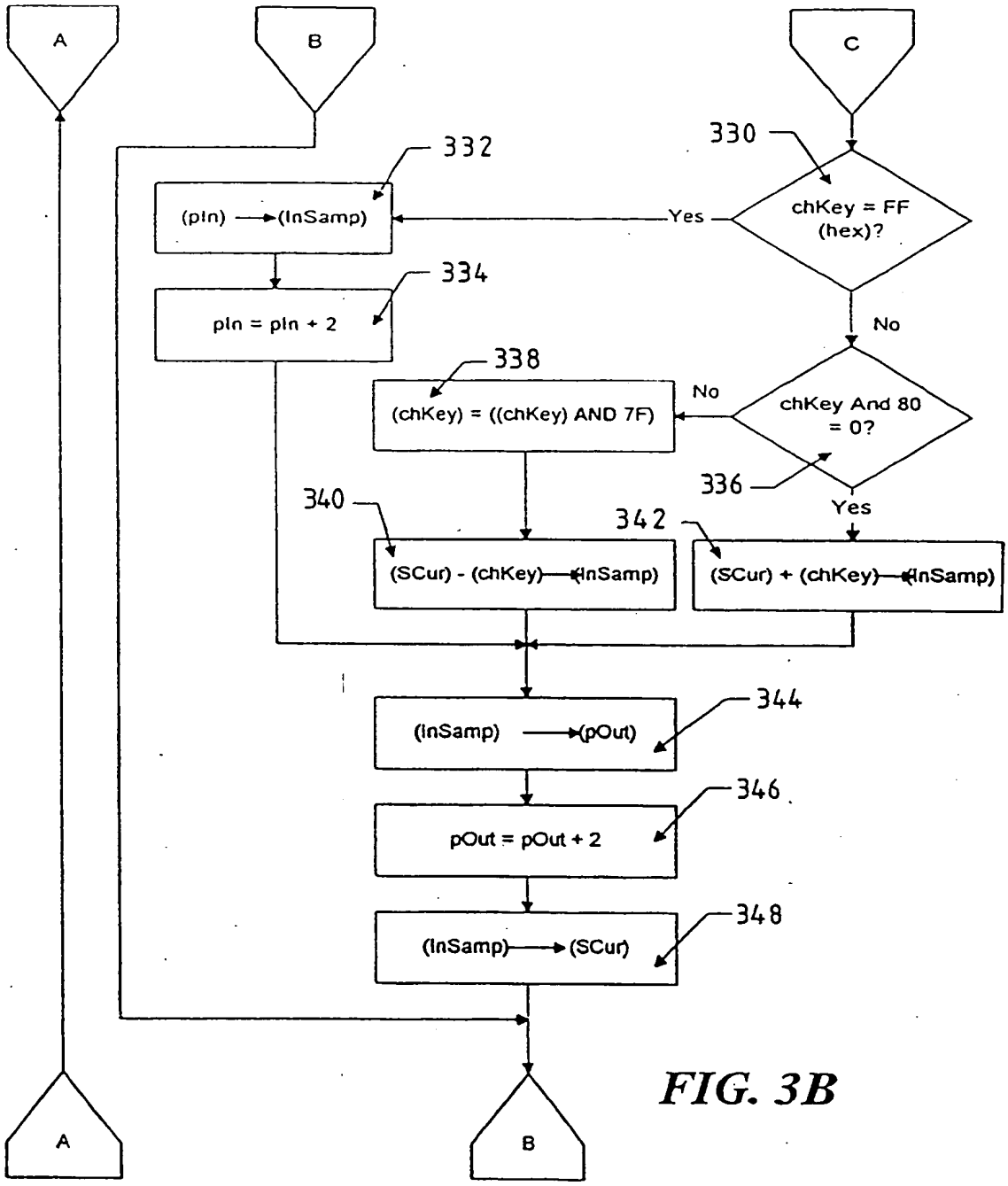
A
B
C
330
chKey = FF (hex)?
Yes
No
332
(pIn) → (InSamp)
334
pIn = pIn + 2
338
(chKey) = ((chKey) AND 7F)
No
chKey And 80 = 0?
336
Yes
340
342
(SCur) - (chKey) → (InSamp)
(SCur) + (chKey) → (InSamp)
344
(InSamp) → (pOut)
346
pOut = pOut + 2
348
(InSamp) → (SCur)
A
B

FIG. 3B

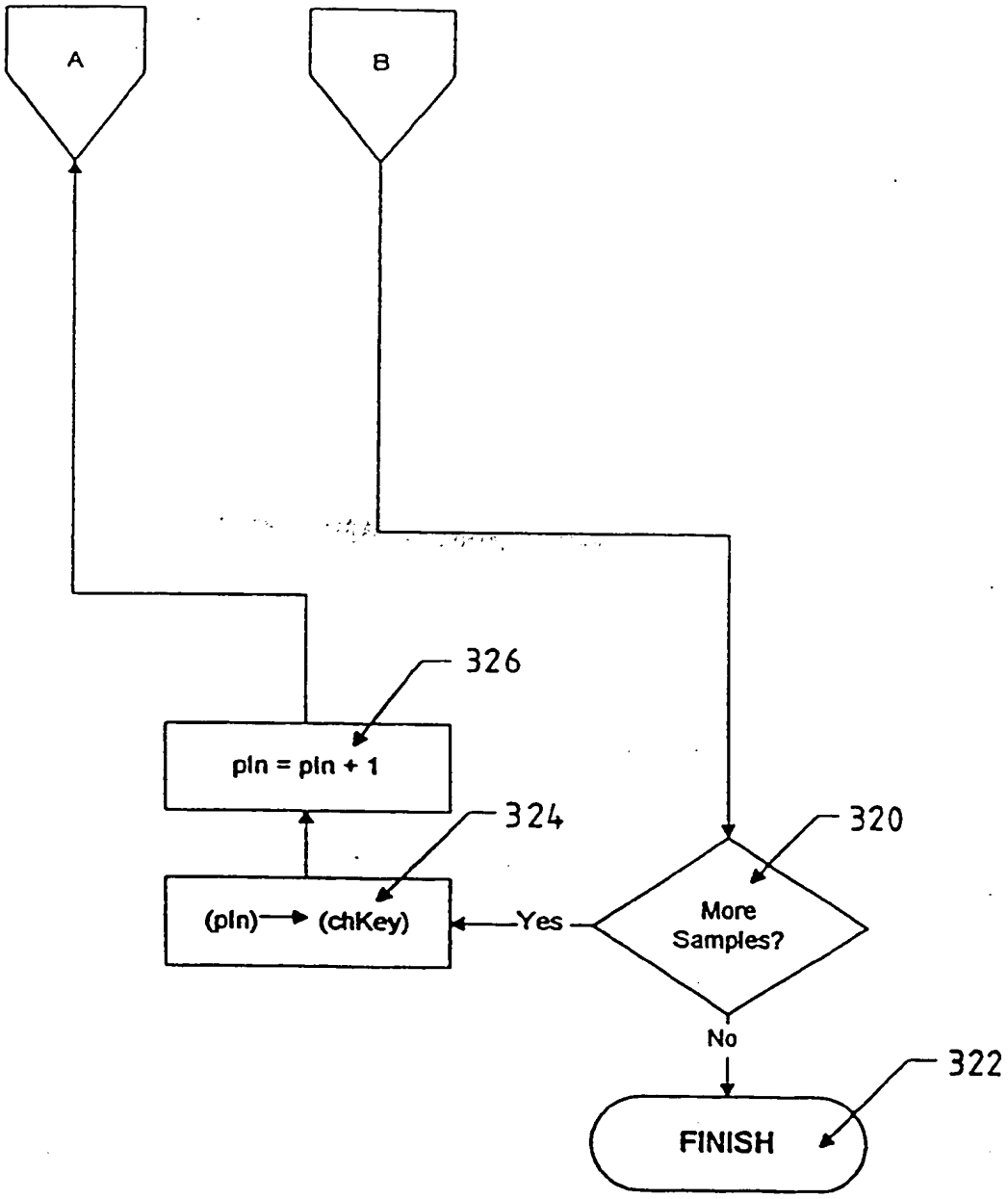
A
B
326
pin = pin + 1
324
(pin) → (chKey)
Yes
320
More Samples?
No
322
FINISH

FIG. 3C

THIS PAGE BLANK (USPTO)

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 661 819 A3

(12) EUROPEAN PATENT APPLICATION

(88) Date of publication A3:
02.02.2000 Bulletin 2000/05

(43) Date of publication A2:
05.07.1995 Bulletin 1995/27

(21) Application number: **94308662.9**

(22) Date of filing: **23.11.1994**

(51) Int. Cl.[7]: **H03M 7/30**, H04B 14/06

(84) Designated Contracting States:
DE FR GB

(30) Priority: **29.12.1993 US 174800**

(71) Applicant:
International Business Machines Corporation
Armonk, N.Y. 10504 (US)

(72) Inventor: **Jones, Ronald Henry**
Delray Beach, Florida 33445 (US)

(74) Representative:
Burt, Roger James, Dr.
IBM United Kingdom Limited
Intellectual Property Department
Hursley Park
Winchester Hampshire SO21 2JN (GB)

(54) Data compression

(57) Digitized audio information is compressed by retrieving a first audio sample from a digitized audio signal and storing it in its entire, uncompressed, form. The next audio sample is then retrieved and a signed difference between the first audio sample value and the next audio sample value is computed. If this difference value can be represented in fewer data segments than would be required to represent the next audio sample, the difference value is stored, rather than the value of the next sample, otherwise the next sample is stored. The invention proceeds in this fashion until the entire PCM audio signal is compressed and stored. The compressed data stream is decompressed to generate an output data stream by retrieving a stored sample and examining it to ascertain whether it is a difference value or a flag which indicates that the following data represents a coded amplitude value. If the sample is a difference value it is digitally added to the previous value (taking into account the sign) and the sum is inserted into the output data stream. The sum value is retained in order to be used to compute the next output value if the following sample is also a difference value. Alternatively, if the sample is a flag value it is discarded and the next sample is retrieved and inserted into the output data stream.

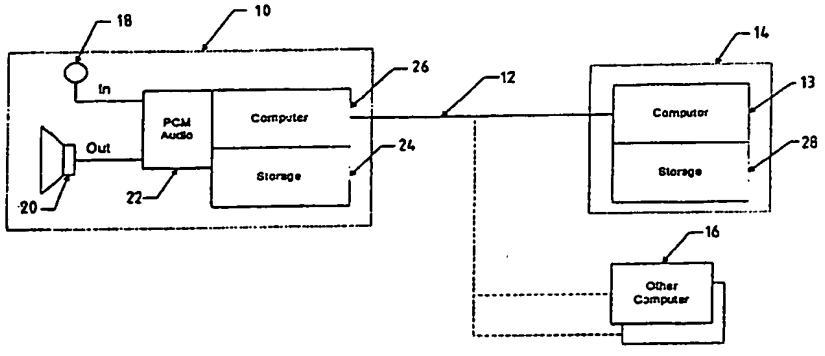

FIG. 1

EP 0 661 819 A3

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 94 30 8662

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X	US 4 783 792 A (AKAGIRI KENZO ET AL) 8 November 1988 (1988-11-08) * column 3, line 44 - column 4, line 49; claims 1,2,5-7; figures 1-3,5 *	1-11	H03M7/30 H04B14/06
A	DE 34 05 184 A (SIEMENS AG) 5 September 1985 (1985-09-05) * page 4, line 1 - page 5, line 5 *	1,6,10, 11	
A	DE 34 03 307 A (SIEMENS AG) 8 August 1985 (1985-08-08) * abstract; claim 1 *	1,6,10	

TECHNICAL FIELDS SEARCHED (Int.Cl.6)

H04B

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	13 December 1999	Nilsson, M

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 94 30 8662

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

13-12-1999

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 4783792 A	08-11-1988	JP 1859994 C	27-07-1994
		JP 5064491 B	14-09-1993
		JP 60251733 A	12-12-1985
		AU 585515 B	22-06-1989
		AU 4401585 A	31-12-1985
		DE 3585150 A	20-02-1992
		EP 0185095 A	25-06-1986
		WO 8505747 A	19-12-1985
		KR 9508107 B	25-07-1995
DE 3405184 A	05-09-1985	NONE	
DE 3403307 A	08-08-1985	NONE	

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

THIS PAGE BLANK (USPTO)